Data Processing Apparatus and Data Processing Method

CROSS REFERENCE TO RELATED APPLICATIONS

This application claims the priority right under 35 U.S.C. 119 of Japanese Patent Application No. 2000-89508 filed on March 28, 2000.

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates to a data processing apparatus for performing a pipeline processing in a plurality of divided stages and, for example, it aims at a data processing apparatus which can be mounted inside a processor.

Related Background Art

With the development of multimedia and communication technique, the enhancement of processing properties of a processor has strongly been desired. Examples of a technique for enhancing the processing properties of the processor include a technique of raising an operation clock frequency, and a technique of performing an arithmetic processing in parallel.

However, when a plurality of operation units are disposed inside the processor and an arithmetic processing is executed in parallel, a circuit scale is enlarged, and the processing may not be performed in time by a wiring delay.

On the other hand, for a recent processor, in order to accelerate execution of an instruction, each instruction is divided into a plurality of stages and subjected to a pipeline processing in many cases. Fig. 1 is a block diagram showing a schematic constitution of a pipeline processor inside the processor, and Fig. 2 is a diagram showing a processing flow.

As shown in Fig. 1, each instruction is divided into five stages A to E and executed in order.

As shown in Fig. 1, each stage is provided with a flip-flop 11 for synchronizing input data, a logic circuit 12, and a multiplexer 13. An output of the multiplexer 13 is inputted to the flip-flop 11 of the next stage.

As shown in Fig. 2, when each instruction is subjected to the pipeline processing, the processor processing properties are enhanced. In order to further enhance the processing properties, however, a plurality of pipeline processing portions are sometimes disposed inside the processor.

Fig. 3 is a block diagram showing an example in which a plurality of pipeline processing portions are disposed inside the processor. An instruction read from an instruction cache (IC) 21 of Fig. 3 is dispatched to an empty pipeline processing portion among six pipeline processing portions (ALU) 24 via an instruction register (IR) 22, and then is executed by the empty pipeline processing portion. Data read out from a register file (RF) 23 in accordance with the instruction is calculated by the pipeline processing portion 24, and the execution result of the instruction is written back to a register file (RF) 23.

Fig. 4 is a block diagram showing a detailed constitution in the vicinity of an input of the pipeline processing portion 24. As shown in Fig. 4, a multiplexer 26 and flip-flop 27 are disposed between the register file 23 and pipeline processing portion 24. Since each pipeline processing portion 24 performs the processing in parallel, a control signal Control is supplied to each multiplexer 26 via a common control line, and each pipeline processing portion 24 performs an arithmetic processing based on the control signal Control.

However, when a plurality of pipeline processing portions are controlled with one control line, with a larger number of pipeline processing portions and a longer wiring length of the control line, fanout load of a control signal increases. In the recent processor, the operation clock frequency is very high. Therefore, there is possibility that the processing in each stage is not in time because of a control signal delay.

In order to reduce the fanout load of the control signal, it is preferable to reduce the wiring length of the control line. However, to enhance the processor processing properties, the number of pipeline processing portions has to be increased, and the wiring length of the control line is necessarily increased.

As another technique for reducing the fanout load of the

control signal, the control signal may be buffered on a tree and supplied to each pipeline processing portion, or a plurality of control signals may be generated beforehand.

Furthermore, in recent years, to develop the processor and ASIC, a technique of arbitrarily combining various prepared function blocks to design LSI has become general. When the designing technique is employed, the combination of function blocks cannot be completely specified. Therefore, it is preferable to preset the fanout load of each signal with an allowance. However, it has heretofore been difficult to set the fanout load of the signal having a critical timing to a value such that erroneous operation is prevented.

SUMMARY OF THE INVENTION

The present invention has been developed in consideration of this respect, and an object thereof is to provide a data processing apparatus which can reduce a fanout load of a control signal for controlling a pipeline.

To achieve the object, there is provided a data processing apparatus configured to perform a pipeline processing in a plurality of divided stages, comprising:

a first pipeline processing portion configured to perform a processing in each stage based on a control signal inputted to each stage;

a first latch portion configured to latch the control signal inputted to each stage with a predetermined clock; and

a second pipeline processing portion, disposed separately from the first pipeline processing portion, configured to perform the processing in each stage based on the control signal latched by the first latch portion.

According to the present invention, instead of directly supplying the control signal to all the pipeline processing portions, the control signal is once latched by the first latch portion and supplied to at least some of the pipeline processing portions, so that the control signal fanout load can be reduced. Therefore, even with a large number of pipeline processing portions, propagation delay of the control signal can be reduced. Moreover,

even if a wiring length of the control line for transmitting the control signal is long, the signal is synchronized with a clock on the way of the wiring, so that the signal is not influenced by wiring delay.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a block diagram showing a schematic constitution of a pipeline processing portion inside a processor.

Fig. 2 is a diagram showing a processing flow of Fig. 1.

Fig. 3 is a block diagram showing an example in which a plurality of pipeline processing portions are disposed inside the processor.

Fig. 4 is a block diagram showing a detailed constitution in the vicinity of an input of the pipeline processing portion of Fig. 3.

Fig. 5 is a block diagram showing one embodiment of a data processing apparatus according to the present invention.

Figs. 6A and 6B are explanatory views of operation of first and second pipeline processing portions.

Fig. 7 is a block diagram showing one example of the data processing apparatus in which a processing result in the first pipeline processing portion can be transmitted to the second pipeline processing portion.

Fig. 8 is a block diagram showing one example of the data processing apparatus in which the processing result in the second pipeline processing portion can be transmitted to the first pipeline processing portion.

Fig. 9 is a diagram showing an example in which the control signal is branched into a plurality of signals in the first and second pipeline processing portions.

Fig. 10 is a diagram showing a constitution in the processor.

Fig. 11 is a block diagram showing an internal constitution of a plural data streams (SIMD) instruction type processor.

Fig. 12 is a diagram showing an example in which the second pipeline processing portion performs a processing a half clock behind the first pipeline processing portion.

Fig. 13 is a diagram showing a detailed constitution of

a latch.

Fig. 14 is a diagram corresponding to Fig. 7, and shows an example in which the second pipeline processing portion performs the processing a half clock behind the first pipeline processing portion.

Fig. 15 is a diagram corresponding to Fig. 8, and shows an example in which the second pipeline processing portion performs the processing a half clock behind the first pipeline processing portion.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

A data processing apparatus of the present invention will concretely be described hereinafter with reference to the drawings. An example of a pipeline processing portion mounted inside a processor will be described hereinafter.

Fig. 5 is a block diagram showing one embodiment of the data processing apparatus according to the present invention. The data processing apparatus of Fig. 5 includes a first pipeline processing portion 1 for executing a processing in five divided stages A to E, a second pipeline processing portion 2 for executing the processing one stage behind the first pipeline processing portion 1, and a plurality of flip-flops (first latch portion) 3 for latching control signals inputted to the respective stages.

Fig. 5 shows an example in which respective separate control signals Control-A, Control-B, Control-C, Control-D, Control-E are supplied to the respective stages, but a common control signal may be supplied to a plurality of stages.

The first and second pipeline processing portions 1, 2 are similarly constituted, and each stage includes a flip-flop 11, logic circuit 12 and multiplexer 13.

The flip-flop 11 latches a previous-stage processing result by a clock CLK dividing the respective stages. Additionally, Fig. 5 shows only one flip-flop 11, but flip-flops 11 are actually disposed for the number of data bits.

The logic circuit 12 performs a predetermined logic and arithmetic operation based on the control signals inputted to the respective stages. Additionally, the logic circuit 12 can

perform the logic and arithmetic operation without using any control signal. The multiplexer 13 selects an output of the logic circuit 12, or an output of the next-stage register file, based on the control signal inputted to each stage.

The flip-flops 3 of Fig. 5 latch the control signals Control-A to E inputted to the respective stages by the clocks CLK dividing the respective stages. Thereby, the respective control signals Control-A to E can be delayed in accordance with a stage processing timing in the second pipeline processing portion 2. This control signal will hereinafter be referred to as a delayed control signal. The delayed control signal is used in the processing in the second pipeline processing portion 2. This delayed control signal is a signal for controlling a processing operation of the first and second pipeline processing portions 1, 2, and concretely a signal for controlling whether or not the pipeline processing is to be stalled.

The control signals Control-A to E are latched by the flip-flops 3 in order to reduce fanout load of the control signals Control-A to E. The control signals Control-A to E are directly supplied to the first pipeline processing portion 1 of Fig. 5 via control lines, while the delayed control signal once latched by the flip-flop 3 is supplied to the second pipeline processing portion 2. Therefore, the delayed control signal supplied to the second pipeline processing portion 2 is not influenced by the fanout load of the control signals Control-A to E on the control line.

Fig. 6 is an explanatory view of the operation of the first and second pipeline processing portions 1, 2, Fig. 6A shows the operation for a case in which the pipeline processing is not stalled, and Fig. 6B shows the operation for a case in which the pipeline processing is stalled.

As shown in Fig. 6, the first pipeline processing portion 1 performs the processing earlier than the second pipeline processing portion 2 by one cycle of the clock CLK. Moreover, when the first pipeline processing portion 1 is stalled for some reason, the processing is interrupted as shown in periods T3, T4 of Fig. 6B. Accordingly, the processing of the second pipeline

processing portion 2 is also interrupted (periods T4, T5).

In the data processing apparatus of Fig. 5, data transfer between the first and second pipeline processing portions 1, 2 is not taken into consideration, but a processing result of one of the first and second pipeline processing portions 1, 2 may be transmitted to the other pipeline processing portion.

For example, Fig. 7 is a block diagram showing one example of the data processing apparatus in which the processing result in the first pipeline processing portion 1 can be transmitted to the second pipeline processing portion 2. The first pipeline processing portion 1 performs the processing earlier than the second pipeline processing portion 2 by one cycle of the clock CLK. Therefore, when the first pipeline processing portion 1 transmits data to the second pipeline processing portion 2, the data to be transmitted needs to be matched with a timing of the second pipeline processing portion 2.

Therefore, in Fig. 7, there is disposed a flip-flop (second latch portion) 14 for latching an output of the logic circuit 12 in the stage C of the first pipeline processing portion 1. The flip-flop 14 latches the output of the logic circuit 12 in synchronization with the clock CLK for dividing the stages, and supplies latched data to the logic circuit 12 in the second pipeline processing portion 2. Since the second pipeline processing portion 2 operates one clock behind the first pipeline processing portion 1, the second pipeline processing portion 2 can receive the processing result in the stage C of the first pipeline processing portion 1, and perform the processing in the stage C.

Additionally, Fig. 7 shows the example in which the processing result of the stage C of the first pipeline processing portion 1 is transmitted to the second pipeline processing portion 2. When the processing result of another stage is transmitted to the second pipeline processing portion 2, the flip-flop 14 may be disposed in the transmitter stage similarly as Fig. 7.

On the other hand, Fig. 8 is a block diagram showing one example of the data processing apparatus in which the processing result in the second pipeline processing portion 2 can be transmitted to the first pipeline processing portion 1.

The second pipeline processing portion 2 operates one clock behind the first pipeline processing portion 1. Therefore, when the processing result in a certain stage of the second pipeline processing portion 2 is transmitted to the first pipeline processing portion 1, the result is transmitted to the subsequent stage. Fig. 8 shows an example in which the processing result in the stage C of the second pipeline processing portion 2 is transmitted to the stage D of the first pipeline processing portion 1.

When the data is transmitted to the first pipeline processing portion 1 from the second pipeline processing portion 2, the first pipeline processing portion 1 is considered to be stalled in some case. In this case, the data to be transmitted has to be held until the first pipeline processing portion 1 resumes the processing.

Therefore, in Fig. 8, there are disposed a flip-flop (third latch portion) 15 for latching the data to be transmitted to the first pipeline processing portion 1 from the second pipeline processing portion 2, and a multiplexer (selector) 16 for selecting either one of an output of the flip-flop 15 and the processing result in the stage C of the second pipeline processing portion 2.

If the first pipeline processing portion 1 is not stalled when the processing result in the stage C of the second pipeline processing portion 2 is obtained, the multiplexer 16 selects the processing result and transmits the result to the stage D of the first pipeline processing portion 1. Moreover, when the processing result in the stage C of the second pipeline processing portion 2 is obtained and the first pipeline processing portion 1 is stalled, the flip-flop 15 latches the processing result in the stage C till the completion of the stall.

Additionally, Fig. 8 shows an example in which the flip-flop 15 for latching the processing result in the stage C of the second pipeline processing portion 2 and the multiplexer 16 are disposed, but the flip-flop 15 and multiplexer 16 of Fig. 8 may be disposed in another stage. Moreover, the flip-flop 14 of Fig. 7 and the flip-flop 15 and multiplexer 16 of Fig. 8 may be disposed.

As described above, in the present embodiment, when a plurality of pipeline processing portions perform processings in parallel, for some of the pipeline processing portions, the processing is performed in each stage based on the delayed control signals Control-A to E generated by once latching the control signals Control-A to E inputted to the respective stages by the flip-flop 3. Therefore, the fanout load of the control signals Control-A to E is reduced, and signal delays of the control signals Control-A to E can be reduced. Moreover, even if a long wiring of the control line for transmitting the control signals Control-A to E is long, the flip-flop 3 is disposed on the way of the wiring and the signal can be synchronized with the clock. Therefore, the wiring length of the control line can set to be longer more than a conventional wiring length.

Furthermore, even with a large number of pipeline processing portions, the corresponding number of flip-flops 3 may be disposed. Therefore, the operation can be stabilized regardless of the number of pipeline processing portions.

In the aforementioned embodiment, the example in which two pipeline processing portions 1, 2 are disposed in the data processing apparatus has been described, but the number of pipeline processing portions and the number of pipeline stages are not particularly limited.

Moreover, Fig. 5 shows the example in which the control signals Control-A to E are latched with the clock CLK for dividing the stages, but the control signals Control-A to E may be latched at a timing other than that of the clock CLK.

Fig. 7 shows the example in which the control signal Control-C inputted to a logic circuit LOGIC-C1 and multiplexer MUX-C1 of the stage C in the left-side pipeline processing portion is latched by the flip-flop 3 and the resulting delayed control signal is supplied to the stage C in the right-side pipeline processing portion, but the control signal Control-C and delayed control signal are sometimes utilized in a plurality of places in the respective pipeline processing portions.

Fig. 9 shows an example in which the control signal outputted from a buffer is branched into a plurality of signals in the first

pipeline processing portion 1, one branched signal is latched by the flip-flop to generate the delayed control signal, and the generated delayed control signal is further branched into a plurality of signals in the second pipeline processing portion 2.

When the number of branched control signals is large in this manner, the buffer and flip-flop are disposed in the course of branching, so that the fanout load of the control signal can be prevented from increasing. Moreover, even when the first and second pipeline processing portions 1, 2 are mounted in positions apart from each other on a die, the flip-flop for latching the control signal is disposed between the pipeline processing portions, so that deviation of the clock from an edge can be reduced.

On the other hand, Fig. 10 is a diagram showing a constitution in the processor. The data is directly supplied to the first pipeline processing portion 1 from an instruction cache 31 via an instruction register 32, and once latched by the flip-flop 3 before supplied to the second pipeline processing portion 2.

The first pipeline processing portion 1 executes the processing one stage before the second pipeline processing portion 2. Therefore, when the data is transmitted to the second pipeline processing portion 2 from the first pipeline processing portion 1, the data is once latched by the flip-flop 3 and timing is adjusted. Conversely, when the data is transmitted to the first pipeline processing portion 1 from the second pipeline processing portion 2, the flip-flop is unnecessary.

The first pipeline processing portion 1 of Fig. 10 includes an integer unit pipeline, load/store unit pipeline, and branch unit pipeline, and the respective pipelines exchange the data with a data cache. Moreover, the second pipeline processing portion 2 includes a floating-point unit pipeline and multimedia unit pipeline.

Additionally, types of pipelines disposed inside the first and second pipeline processing portions 1, 2 are not especially limited, or are not limited to those of Fig. 10.

For example, the integer unit pipeline and load/store unit pipeline may be disposed in the second pipeline processing portion

2, and the floating-point pipeline and multimedia unit pipeline may be disposed in the first pipeline processing portion 1.

On the other hand, Fig. 11 is a block diagram showing an internal constitution of a plural data stream (SIMD) instruction type processor. As shown in Fig. 11, a plurality of arithmetic and logic units (ALU) 24 are disposed inside the first and second pipeline processing portions 1, 2. The data passed from the instruction cache 31 via the instruction register 32 is supplied to the first pipeline processing portion 1 as it is, and once latched by the flip-flop 3 before supplied to the second pipeline processing portion 2. Moreover, the first pipeline processing portion 1 performs the processing one stage before the second pipeline processing portion 2. Therefore, when the data is transmitted to the second pipeline processing portion 2 from the first pipeline processing portion 1, the data is once latched by the flip-flop 3. Conversely, when the data is transmitted to the first pipeline processing portion 1 from the second pipeline processing portion 2, the flip-flop is unnecessary.

Additionally, Fig. 5, and the like show the example in which the second pipeline processing portion 2 performs the processing one stage (one clock) behind the first pipeline processing portion 1, but the second pipeline processing portion 2 may perform the processing with a delay amount other than one stage.

Fig. 12 shows an example in which the second pipeline processing portion 2 performs the processing a half clock behind the first pipeline processing portion 1. In Fig. 12, a latch 3a is disposed instead of the flip-flop 3 of Fig. 5, and each latch 3a latches the control signals Control-A to C with a falling edge of the clock CLK for dividing the stages, and supplies the latched delayed control signal to the second pipeline processing portion 2.

Fig. 13 is a diagram showing a detailed constitution of the latch 3a. As shown in Fig. 13, different from the flip-flop, the latch 3a outputs the data inputted to an input terminal D via a terminal Q when a terminal E is at a high level, and holds logic of the input terminal D immediately before the terminal E when the terminal E reaches a low level.

On the other hand, Fig. 14 corresponds to Fig. 7, and shows an example in which an operation result of the logic circuit LOGIC-C1 in the stage C of the first pipeline processing portion 1 is latched by the latch 3a at a falling edge of the clock CLK and the latched result is supplied to the stage C of the second pipeline processing portion 2.

On the other hand, Fig. 15 corresponds to Fig. 8, and shows an example in which the second pipeline processing portion 2 transmits the data to the first pipeline processing portion 1. Two-stage latches 3a connected in tandem, and multiplexer 13 are disposed inside the second pipeline processing portion 2. The first-stage latch 3a latches an output of the multiplexer 13 when the clock CLK is at the high level, and the second-stage latch 3a latches the output of the first-stage latch 3a when the clock CLK is at the low level. The output of the first-stage latch 3a is transmitted to the first pipeline processing portion 1.

Moreover, the multiplexer 13 selects either one of the output of the second-stage latch and the data from the stage B in accordance with the output of the latch for latching the data at the falling edge of the clock CLK.